Deep Operator Networks Assignment

CE397 and CSE393: Scientific Machine Learning

---

1 Learning the Derivative Operator

Design a DeepONet to learn the derivative operator $\mathcal{G}$ that maps input functions $u(x)$ to their derivatives:

$$\mathcal{G}[u](x) = \frac{du}{dx}, \quad x \in [-1, 1]$$

Consider a family of sine functions as the input function space:

$$u(x) = A\sin(\omega x + \phi)$$

The analytical derivative is:

$$\frac{du}{dx} = A\omega\cos(\omega x + \phi)$$

a) Design a DeepONet architecture with:

- **Branch Network:** Input function values $u(x_j)$ at $m = 100$ sensor locations, two hidden layers with 32 neurons each, Tanh activations, output $p = 50$ basis coefficients $b_k(u)$
- **Trunk Network:** Input query location y, two hidden layers with 32 neurons each, Tanh activations, output $p = 50$ basis functions $t_k(y)$
- **Output:** $\mathcal{G}(u)(y) = \sum_{k=1}^{p} b_k(u) \cdot t_k(y) + b_0$

b) Generate $N = 2000$ training functions:

- Amplitude: $A \sim \text{Uniform}(0.5, 2.0)$
- Frequency: $\omega \sim \text{Uniform}(1, 5)$
- Phase: $\phi \sim \text{Uniform}(0, 2\pi)$
- Sample each function at $m = 100$ sensor locations uniformly in $[-1, 1]$
- Compute analytical derivatives: $\frac{du}{dx} = A\omega\cos(\omega x + \phi)$

c) Implement the data loss function:

$$\mathcal{L}_{data} = \frac{1}{N \cdot P}\sum_{i=1}^{N}\sum_{j=1}^{P}\left|\mathcal{G}_\theta(u^{(i)})(y_j) - \frac{du^{(i)}}{dx}(y_j)\right|^2$$

d) Train using 80-10-10 train-validation-test split, batch size 128, Adam optimizer ($lr = 10^{-2}$), ReduceLROnPlateau scheduler (patience=50, factor=0.5), early stopping (patience=100), maximum 500 epochs.

e) Analyze the learned basis functions:

- Visualize the first 8 trunk network basis functions $t_k(y)$
- Test specific cases: $u(x) = \sin(x)$, $u(x) = \sin(2x)$, $u(x) = 2\sin(3x + \pi/4)$
- Plot predictions vs. analytical derivatives for these cases
- Explain how the branch network encodes frequency and amplitude information

f) Evaluate generalization:

- Compute relative L^2 error on test set: $\dfrac{\left\| u'_{pred} - u'_{true} \right\|_2}{\| u'_{true} \|_2}$
- Plot predictions vs. true derivatives for 6 test samples
- Report mean and standard deviation of test errors

2 1D Advection-Diffusion Equation

Learn the solution operator for the steady-state advection-diffusion equation:

$$\nu \frac{d^2 u}{dx^2} - c\frac{du}{dx} = f(x), \quad x \in [0, 1]$$

with Dirichlet boundary conditions $u(0) = u(1) = 0$, diffusion coefficient $\nu = 0.01$, and advection velocity $c = 1.0$.

a) Generate training data:

- Create $N = 1000$ source functions using Fourier series:

$$f(x) = \sum_{n=1}^{4} c_n \sin(n\pi x), \quad c_n \sim \mathcal{N}(0, (1/n)^2)$$

- For each $f(x)$, solve the BVP numerically using finite differences with $n_x = 100$ grid points
- Discretize using central differences for diffusion and upwind for advection:

$$\nu \frac{u_{i-1} - 2u_i + u_{i+1}}{\Delta x^2} - c\frac{u_{i+1} - u_i}{\Delta x} = f_i$$

b) Design a DeepONet with:

- Branch network: source function values at 100 points, three hidden layers (128 neurons), Tanh, output 100 coefficients
- Trunk network: spatial coordinate, three hidden layers (128 neurons), Tanh, output 100 basis functions

c) Normalize targets using training statistics: $\tilde{u} = (u - \mu_u)/\sigma_u$.

d) Train using 80-20 split, batch size 64, Adam ($lr = 10^{-3}$, weight decay 10^{-5}), cosine annealing scheduler, 3000 epochs.

e) Evaluate: compute relative L^2 error, plot 6 test cases (source and solution), visualize 8 learned basis functions.

f) Compare forward pass time vs. finite difference solver time for a single evaluation.

3 Physics-Informed DeepONet

Develop a physics-informed DeepONet for the parametric advection-diffusion equation:

$$\nu \frac{d^2u}{dx^2} - c\frac{du}{dx} = f(x), \quad x \in [0, 1]$$

with Dirichlet BCs $u(0) = u(1) = 0$ with diffusion coefficient $\nu = 0.01$, and advection velocity $c = 1.0$. .

a) Use the generatee dataset from the previous question with $N = 1000$ training samples.

b) Design a PI-DeepONet:

- Branch network: concatenate $[f(x_1), \ldots, f(x_m), \nu, c]$, three hidden layers (64 neurons), Tanh, output 50 coefficients
- Trunk network: spatial coordinate x, three hidden layers (64 neurons), Tanh, output 50 basis functions
- Enable automatic differentiation to compute $\frac{du}{dx}$ and $\frac{d^2u}{dx^2}$

c) Implement physics-informed loss:

$$\mathcal{L}_{data} = \frac{1}{NP}\sum_{i,j} |\mathcal{G}_\theta(f^{(i)}, \nu^{(i)}, c^{(i)})(x_j) - u_{true}^{(i)}(x_j)|^2$$

$$\mathcal{L}_{physics} = \frac{1}{NP}\sum_{i,j} \left| \nu^{(i)} \frac{d^2\mathcal{G}_\theta}{dx^2}(x_j) - c^{(i)}\frac{d\mathcal{G}_\theta}{dx}(x_j) - f^{(i)}(x_j) \right|^2$$

$$\mathcal{L}_{BCs} = \frac{1}{N}\sum_{i} \left[|\mathcal{G}_\theta(\cdot)(0)|^2 + |\mathcal{G}_\theta(\cdot)(1)|^2 \right]$$

$$\mathcal{L} = \mathcal{L}_{data} + \lambda_{physics}\mathcal{L}_{physics} + \lambda_{BCs}\mathcal{L}_{BCs}$$

e) Train with $\lambda_{physics} = 1.0$, $\lambda_{BCs} = 10.0$, batch size 32, Adam ($lr = 10^{-3}$, weight decay 10^{-4}), ReduceLROnPlateau (patience=100), early stopping (patience=200), 1000 epochs.

f) Track all loss components during training. Plot total, data, physics, and BC losses (log scale). Create bar chart of final losses.

g) Evaluate on test set:

- Average relative L^2 error
- Average physics residual: $|\nu \frac{d^2u_{pred}}{dx^2} - c\frac{du_{pred}}{dx} - f|$
- Average boundary error: $|u_{pred}(0)| + |u_{pred}(1)|$

h) Compare answer with Q2